친지인 최종 Proposal

(Chin-ji-in, 櫬地人)



오픈소스SW프로젝트 01분반 – TEAM 9

이창민 서정현 한태균 임우섭 정석우

# 목차

1. **서론**
   1. 프로젝트 소개 및 필요성
      1. 친지인이란?
      2. 천지인 키보드 특징
      3. 천지인 오타 특징
   2. 기존 기술 조사 및 차별성
      1. 스마트폰에 내장된 자동완성 기능
      2. Peter Norvig
      3. Symspell
      4. 한글 오타 교정
2. **본론**
   1. 개발 방법
      1. 개발 언어
      2. 알고리즘
      3. 업무 분담
3. **결론**
   1. 프로젝트 의의

# 프로젝트 배경

## 친지인(櫬地人)

천지인 키보드에서 오타를 줄이기 위해 고안된 프로젝트이다. 단어 ‘천지인’에서 아래아(ㆍ)가 탈락된 오타로 이름이 지어졌으며, 한자는 무궁화 나무 친 ‘櫬’을 사용하여 한글 창제 원리를 기반으로 한 천지인 키보드가 우리나라 사람들에게 보다 쉽게 사용될 수 있길 바라는 마음을 담았다.

## 천지인 키보드 특징

가장 흔히 사용되는 두벌식(qwerty) 키보드와 비교해보면 천지인 키보드는 기본적으로 필요한 자판이 문자 자판 10개와 공백 자판으로 총 11개인 반면, 두벌식 키보드는 기본적으로 문자 자판 26개와 공백 자판, Shift 자판이 필요하므로 총 28개의 자판을 가진다. 따라서 천지인 키보드는 자판이 넓지만 타수가 많아지고, 두벌식 키보드는 자판은 좁지만 타수는 적어진다. 이에 천지인은 두벌식 보다 한 손으로 입력하기 용이하다는 장점을 가진다.

많은 타수를 입력해야 하는 천지인 키보드 사용자는 두벌식 키보드 사용자보다 빠른 입력이 요구되기 때문에 넓은 자판에도 불구하고 자판을 잘못 누르는 경우가 종종 발생한다. 특히 자판 간의 순서 바뀜, 자판 하나 안눌림, 자판 하나 더눌림과 같은 오타는 자판의 크기와 크게 관련없이 발생하기 때문에 **천지인에서의 오타가 두벌식에서보다 빈번히 발생**하는 것으로 조사되었다.

## 천지인 오타 특징

실제로 팀원들이 천지인 키보드를 사용하여 오타 수정없이 노래 총 12곡의 가사들을 입력해 보고 통계를 내보았는데, 총 1592단어에서 260개의 단어에서 오타가 발생했으며, 이는 16%의 비율이다. 오타의 원인으로는 Insert(자판 한개 추가 입력)이 30%로 가장 빈번했으며, Replace(자판 한개 다른 자판 입력)이 22%, Deletion(자판 한개 입력 안함)이 20%, Transpose(자판 한 쌍 순서 바뀜)가 7%를 차지했으며, 이외에 한 단어에서 두가지 이상의 오타가 일어난 경우는 20%였다.

이 데이터셋에서 중요한 것은 오타의 내용이었다. 천지인 키보드의 오타는 두벌식 키보드에서는 발생하기 힘든 오타가 종종 발견되기도 했으며, 두벌식 키보드에서 쉽게 발생할 법한 오타가 천지인 키보드에서는 가능성이 적은 경우도 많았다.

예를 들면, ‘고양이’라는 단어에서 양과 이 사이에 스페이스바를 누르지 않으면 ‘고야미’라는 결과가 나오고, 양에서 아래아(ㆍ) 하나를 ‘ㅣ’보다 먼저 누르면 ‘고어ㆍㅇ이’ 라는 결과가 나오게 되는데 이는 모두 두벌식에서는 나타나기 힘든 오타이다.

또한, 천지인 키보드에는 두 번 이상 눌러야 입력할 수 있는 자음과 모음(ㅋ, ㄲ, ㅁ, ㅏ, ㅠ 등)이 많기 때문에 이와 같은 음소가 실수로 추가되었다고 보는 것은 우선 순위가 낮다. 예를 들어 ‘맴수’라는 입력이 있으면 ‘매수’보다는 ‘맹수’로 교정해주는 것이 옳다는 것이다.

위와 같은 특징들이 존재하기 때문에 **천지인 전용의 오타 교정 알고리즘이 필요**하다.

# 기존 기술 조사 및 차별성

스마트폰에서 쉽게 볼 수 있는 자동 완성 기능과, 영어 오타 교정 알고리즘으로 유명한 **Peter-Norvig Algorithm**, **symspell** 알고리즘을 조사하고, symspell에 두벌식 키보드 자판의 물리적 거리 가중치를 적용한 **customed-symspell**또한 조사하였다. 그리고 기존에 개발되어 있는 한글 오타교정을 찾아보았다.

## 자동 완성 기능

기존에 알고 있던 단어와 사용자가 자주 입력했던 단어를 기반으로, 현재 입력중인 단어를 완성 또는 대체할 단어들을 추천해준다.

사용자가 입력한 단어를 기억하기 때문에 사전에 없는 단어도 수정해주는 것이 장점이지만, **오타교정 목적으로 사용하기에는 한계가 존재**한다. 가장 먼저, 위에서 언급했던 특수한 천지인 오타(‘고야미’, ‘고어ㆍㅇ이’ 등)를 전혀 잡지 못하였다. 그리고 ‘노양이’, ‘비아노’등의 오타를 못 고쳐주는 것으로 보아 이 기능은 오타교정보다는 자동완성을 목적 하에 구현되었기에 단어 초반의 오타를 잘 잡아주지 못하는 것으로 보인다. 또한 이 기능은 여러 교정이 가능한 상황에서는 단어의 빈도수만을 기반으로 우선순위를 두기 때문에 보다 가능성 높은 오타를 추천해주지 않는다.

위와 같은 한계가 존재하므로 **‘친지인’ 프로젝트는 자동완성기능과 차별성을 지닌다.**

Peter Norvig 알고리즘

Peter Norvig 알고리즘은 미국의 컴퓨터 과학자 Peter Norvig에 의해 고안된 가장 대표적인 영어 오타 교정 알고리즘이다.

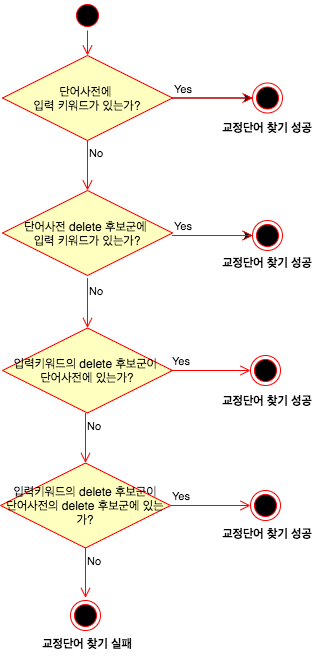
이 알고리즘은 우선 교정 사전으로 활용할 텍스트 파일을 구축하는데, 텍스트 파일은 단어와 그 단어의 빈도 수를 저장한다. 이 빈도 수는 교정 단어 후보 간의 우선순위를 정할 때 사용된다.

어떤 단어를 입력 받았을 때, 단어가 사전에 있다면 그대로 반환한다. 사전에서 찾지 못한 경우, 입력 단어에서 delete, insert, transpose, replace 네가지 유형에 대하여 편집 거리가 1인 교정후보들을 모두 구한다. 예를 들어 ofen이라는 단어가 있다고 하면 delete군은 [fen, oen, ofn, ofe], insert군은 [aofen, bofen, cofen, … ofenz], transpose군은 [foen, oefn, ofne], replace군은 [afen, bfen, … , open, … opez]를 만들 수 있다. 위의 후보들 중에서 사전에 존재하는 단어만을 모아서 최종 후보를 정한 뒤, 사전에 저장되어 있는 단어의 빈도 수를 기반으로 입력 단어를 교정해준다.

만약 편집 거리가 1인 교정후보들 중 사전에 존재하는 단어가 없을 경우 이 교정후보들로부터 각각 편집 거리가 1인 교정후보들을 생성한다. 즉, 이는 입력 단어에 대해 편집 거리가 2인 교정후보들이 된다. 마찬가지로 이 후보들 중 사전에 존재하는 단어를 빈도 수 기반으로 입력 단어를 교정해주며, 거리가 2인 교정후보도 존재하지 않을 경우엔 입력 단어를 그대로 반환해준다.

가장 직관적인 알고리즘이지만 동작이 느리다는 단점이 존재한다. 영어 기준으로 입력 길이가 n인 단어에 대하여 편집 거리가 1인 교정 후보의 개수는 54n + 25개이며, 편집 거리가 2인 교정후보의 개수는 대략 개나 되기 때문이다.

Symspell 알고리즘 - <https://github.com/MighTguY/customized-symspell>

Peter Norvig의 높은 연산비용 문제를 해결한 알고리즘이 Symspell 알고리즘이다. 이 알고리즘은 우선 단어사전 파일을 읽어서 모든 단어에 대한 delete후보군을 계산해 놓는다.

그 후 단어를 입력받았을 때 단어 사전에 있는 단어면 그대로 반환한 다. 없다면 단어 사전으로부터 미리 계산해놓은 단어사전 delete 후보군에 입력 단어가 있는지 확인한다. 있다면 사전에서 해당되는 단어를 반환한다. 단어사전 delete 후보군에서 찾지 못했다면 입력 키워드의 delete 후보군을 구한다. 이 각각의 후보군을 단어사전에서 찾아보고 찾았다면 해당 단어를 반환한다. 여기서도 찾지 못했다면 입력 키워드의 delete 후보군 각각을 단어사전 delete 후보군에서 찾아보며, 찾았다면 해당 단어를 반환한다. 못 찾았다면 교정에 실패한 것으로 원래 입력 키워드를 반환한다.

Peter Norvig 알고리즘과 달리 Delete 후보군만을 사용하며 미리 사전 Delete 후보군을 계산해 놓기 때문에 수행 속도가 훨씬 개선되었다.

# 한글 오타 교정

우선 찾아본 결과, 한글 오타 교정 프로젝트는 그렇게 많지 않았다. [hunspell-dict-ko](https://github.com/spellcheck-ko/hunspell-dict-ko)과 [py-hanspell](https://github.com/ssut/py-hanspell)과 같은 프로젝트를 찾아볼 수 있었는데, 이런 한글 오타 교정 알고리즘은 모두 자모를 두벌식 기준으로 분리한다. 따라서 천지인의 특수한 오타를 교정할 수 있는 천지인 입력 오타 교정 알고리즘은 현재 찾아볼 수 없었기 때문에 **‘친지인’ 프로젝트는 차별성을 지닌다.**

# 개발 방법

소스 코드 버전 관리를 위해 Git을 사용한다. 팀의 Git Repository주소는 다음과 같다. <https://github.com/tjwjdgus12/ChinJiIn>

# 아 그리고 막 에자일 방법론 이런 얘기도 써야 할까?

## 개발 언어

프로젝트 특성상 문자열을 많이 다루어야 하므로 문자열에 대한 편리한 기능들이 많은 Python3를 개발 언어로 선택했다.

## 알고리즘

# 이해하고 쓸 예정

## 역할 분담

# 팀장: 이창민

# 쩌리들: 서정현 임우섭 한태균 정석우

# 이거 우리 어떻게 써?

# 프로젝트 의의

# …? 여기 뭐 들어가야 할까? 이거 그냥 없앨까 뭐 결론같은말 없나